

[illegible]

3

Sy

MT

MT
MT

MT

MT

MT
MTMT
MTMT
MTMT
MTMT
MTMT
MT

MT

MT

MT
MT

MI
MT

MT
MT

MT
MT

MT
MT

MT
MT

MT

111

227

MT

MT
MT

MT

MT

M1
M1MT
MT

M1

M1

10

1

10

100

10

1

```

MM      MM      TTTTTTTTTT  HH      HH  HH      HH  LL      000000      GGGGGGGG
MM      MM      TTTTTTTTTT  HH      HH  HH      HH  LL      000000      GGGGGGGG
MMM     MMM     TT      HH      HH  HH      HH  LL      00      00  GG
MMM     MMM     TT      HH      HH  HH      HH  LL      00      00  GG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG
MM      MM      TT      HHHHHHHHHH  HHHHHHHHHH  LL      00      00  GG
MM      MM      TT      HHHHHHHHHH  HHHHHHHHHH  LL      00      00  GG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG  GGGGGG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG  GGGGGG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG      GG
MM      MM      TT      HH      HH  HH      HH  LL      00      00  GG      GG
MM      MM      TT      HH      HH  HH      HH  LL      000000      GGGGGG
MM      MM      TT      HH      HH  HH      HH  LL      000000      GGGGGG

```

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SSSSSS
LL          II             SSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LLLLLLLLLLLL IIIIII          SSSSSSSS
LLLLLLLLLLLL IIIIII          SSSSSSSS

```


(2) 60
(3) 80
(4) 271
(5) 357
(6) 400
(7) 444

HISTORY ; Detailed Current Edit History
DECLARATIONS ; Declarative Part of Module
MTH\$HLOG - Standard H-Floating LOG
MTH\$HLOG10 - Standard H Floating Common logarithm
MTH\$HLOG2 - Standard H Floating Common logarithm
MTH\$HLOGHLOG10_R8 - Special HLOG/HLOG10 routines

```
0000 1      .TITLE  MTH$HLOG      ; Floating Point Natural and Common
0000 2      ;                               ; Logarithm Functions (HLOG, HLOG10)
0000 3      .IDENT /2-005/      ; File: MTHHLOG.MAR Edit: PDG2005
0000 4      ;
0000 5      ;*****
0000 6      ;
0000 7      ; COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      ; ALL RIGHTS RESERVED.
0000 10     ;
0000 11     ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     ; ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     ; INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     ; COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     ; OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     ; TRANSFERRED.
0000 17     ;
0000 18     ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     ; CORPORATION.
0000 21     ;
0000 22     ; DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     ; SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     ;
0000 25     ;*****
0000 26     ;
0000 27     ;
0000 28     ;
0000 29     ; FACILITY: MATH LIBRARY
0000 30     ;++
0000 31     ; ABSTRACT:
0000 32     ;
0000 33     ; MTH$HLOG and MTH$HLOG10 are functions which return the H floating point
0000 34     ; natural or common logarithm of their H floating point argument. The call
0000 35     ; is standard call-by-reference. MTH$HLOG_R8 and MTH$HLOG10_R8 are special
0000 36     ; routines which are the same as MTH$HLOG and MTH$HLOG10 except that a
0000 37     ; faster non-standard JSB call is used with the argument in R0 through R3
0000 38     ; and no registers are saved.
0000 39     ;
0000 40     ;--
0000 41     ;
0000 42     ; VERSION: 1
0000 43     ;
0000 44     ; HISTORY:
0000 45     ; AUTHOR:
0000 46     ;      John A. Wheeler, 24-Sep-1979.
0000 47     ;
0000 48     ; MODIFIED BY:
0000 49     ;
0000 50     ;
0000 51     ; VERSION: 2
0000 52     ;
0000 53     ; HISTORY:
0000 54     ; AUTHOR:
0000 55     ;      Bob Hanek, 23-Jun-1981.
0000 56     ;
0000 57     ;
```


MTH\$HLOG
2-005

1 2
; Floating Point Natural and Common

16-SEP-1984 01:36:48 VAX/VMS Macro V04-00
6-SEP-1984 11:25:02 [MTHRTL.SRC]MTHHLOG.MAR;1

Page 2
(1)

0000 58 ;

MTH\$
Sym
ACM
ERR
ERR
HLOC
HLOC
HLOC
HLOC
H-IL
H-LI
H-LI
H-LI
LN
LN
LOG
LOG
LOG
LOG
LONG
MTH\$
MTH\$
MTH\$
MTH\$
MTH\$
MTH\$
MTH\$
MTH\$
MTH\$
NEG
X

PSE

_MT

Pha

Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass
The

MTH\$HLOG
2-005

J 2
; Floating Point Natural and Common
HISTORY ; Detailed Current Edit History

16-SEP-1984 01:36:48 VAX/VMS Macro V04-00
6-SEP-1984 11:25:02 [MTHRTL.SRC]MTHHLOG.MAR;1

Page 3
(2)

```
0000 60      .SBTTL HISTORY ; Detailed Current Edit History
0000 61
0000 62
0000 63
0000 64 : Edit History for Version 1 of MTH$HLOG
0000 65
0000 66 : 1-001 - Adapted from MTH$GLOG version 1-001. JAW 24-Sep-1979
0000 67 : 1-002 - Reorder additions after POLYH for greater accuracy. JAW
0000 68 : 16-Jan-1980.
0000 69
0000 70
0000 71 : Edit History for Version 2 of MTH$HLOG
0000 72
0000 73 : 2-001 - Add MTH$HLOG2. RNH 08-Aug-1981
0000 74 : 2-002 - Correct entry logic for JSB entries. Use G^ addressing for
0000 75 : externals. SBL 24-Aug-1981
0000 76 : 2-003 - Changed MTH$$AB ALOG to MTH$$AB ALOG_V. RNH 29-Sep-81
0000 77 : 2-004 - Eliminated symbolic short literals. RNH 15-Oct-81
0000 78 : 2-005 - Changed H_FHI to the global symbol MTH$$AB_H_FHI. PDG 3-Nov-81
```



```
0000 80 .SBTTL DECLARATIONS ; Declarative Part of Module
0000 81
0000 82
0000 83 : INCLUDE FILES: MTHJACKET.MAR
0000 84
0000 85 : EXTERNAL SYMBOLS:
0000 86 .DSABL GBL
0000 87 .EXTRN MTH$K_LOGZERNEG ; Error code
0000 88 .EXTRN MTH$$SIGNAL ; Math signal routine
0000 89 .EXTRN MTH$$AB_ALOG_V ; Table of byte offsets
0000 90
0000 91 : EQUATED SYMBOLS:
0000 92
000041FC 0000 93 ACMASK = ^M<IV, R2, R3, R4, R5, R6, R7, R8>
0000 94 ; Register save mask and IV enable
0000 95
0000 96
0000 97 : MACROS: none
0000 98
0000 99 : PSECT DECLARATIONS:
0000 100
00000000 0000 101 .PSECT _MTH$CODE PIC,SHR,LONG,EXE,NOWRT
0000 102 ; Program section for math routines
0000 103
0000 104 : OWN STORAGE: none
0000 105
0000 106 : CONSTANTS:
0000 107
0000 108
0000 109
0000 110
0000 111 : The H_FHI table is accessed by an index obtained from the MTH$$AB_ALOG_V
0000 112 : table. The MTH$$AB_ALOG_V table is located in MTHALOG.MAR. Indices
0000 113 : between 0 and 13 inclusive are used to access entries 0 through 13
0000 114 : respectively. For these indices, the first three items of the
0000 115 : corresponding entry are FHI, LN_FHI_LO and LN_FHI_HI. The last two
0000 116 : items for these entries are not used. Indices between 14 and 27
0000 117 : inclusive access entries 13 through 0 respectively. For these indices,
0000 118 : the last three items in the corresponding entry are LN_FHI_HI, LN_FHI_LO
0000 119 : and FHI. The first two items for these entries are not used.
0000 120
0000 121
0000 122 MTH$$AB_H_FHI::
0000 123 : Entry 0
0000 124 .OCTA ^X00000000000000000000000002000DA9F4001 ; .18539905548095703125000000000000
0000 125 .OCTA ^X421AE3E330BE1E1611FD7FA0396B3FE7 ; .18243440203202331412672562181589
0000 126 .OCTA ^X0000BE807D885E853F2D08F13C144000 ; .61734035439402633689170117061398
0000 127 .OCTA ^X6A77662D008D1E1611FD7FA0396B3FE7 ; .18243440203202331412520490863424
0000 128 .OCTA ^X00000000000000000000000003C0014294000 ; .53937709331512451171875000000000
0000 129 : Entry 1
0000 130 .OCTA ^X000000000000000000000000086009E3A4001 ; .16180804967880249023437500000000
0000 131 .OCTA ^X1A1651666C92CAD4A248A6A11FACBFE8 ; -.33489709905478748530671578417454
0000 132 .OCTA ^X0000EDD09CDBD55BF8475616ECCA3FFF ; .48124060168453993340622550742636
0000 133 .OCTA ^X9528BBCF7973CAD4A248A6A11FACBFE8 ; -.33489709905478748530752865989701
0000 134 .OCTA ^X00000000000000000000000009E003C6C4000 ; .61801618337631225585937500000000
0000 135 : Entry 2
0000 136 .OCTA ^X0000000000000000000000000A80074D14001 ; .14563241004943847656250000000000
```


076616CA	3725BE56	BF739FDC	0CB13FE6	00B0	137	.OCTA	^X076616CA3725BE56BF739FDC0CB13FE6	:	.78200201258022195288972178042032
00004A10	EC326891	34C3FF15	80EF3FFF	00C0	138	.OCTA	^X00004A10EC32689134C3FF1580EF3FFF	:	.37591551367694667405015246963373
EDD7A91E	2104BE56	BF739FDC	0CB13FE6	00D0	139	.OCTA	^XEDD7A91E2104BE56BF739FDC0CB13FE6	:	.78200201258022195288623080689775
00000000	00000000	0000F200	5F914000	00E0	140	.OCTA	^X00000000000000000000F2005F914000	:	.68666034936904907226562500000000
				00F0	141	: Entry 3			
00000000	00000000	00003A00	575A4001	00F0	142	.OCTA	^X000000000000000000003A00575A4001	:	.13412204940850219726562500000000
67A2E43C	EAD688A	1F5CE019	C5A7BFE4	0100	143	.OCTA	^X67A2E43CEAD688A1F5CE019C5A7BFE4	:	.33007801045908702818319461233003
0000B570	E806A316	2F923DC8	2CA03FFF	0110	144	.OCTA	^X0000B570E806A3162F923DC82CA03FFF	:	.29358002218568181002591295133470
B2179E72	1475688B	1F5CE019	C5A7BFE4	0120	145	.OCTA	^XB2179E721475688B1F5CE019C5A7BFE4	:	.33007801045908702818483500041738
00000000	00000000	0000EA00	7DBD4000	0130	146	.OCTA	^X00000000000000000000EA007DBD4000	:	.74558955430984497070312500000000
				0140	147	: Entry 4			
00000000	00000000	00004400	423B4001	0140	148	.OCTA	^X000000000000000000004400423B4001	:	.12587168216705322265625000000000
C62C8D33	F40BBE6D	B8911FC4	ECC13FE5	0150	149	.OCTA	^XC62C8D33F40BBE6DB8911FC4ECC13FE5	:	.71705201262076255106096380347282
0000D1A0	84893746	EDC5E4C2	D73A3FFE	0160	150	.OCTA	^X0000D1A084893746EDC5E4C2D73A3FFE	:	.23009279937625369424115235258925
250B3ED9	E150BE6D	B8911FC4	ECC13FE5	0170	151	.OCTA	^X250B3ED9E150BE6DB8911FC4ECC13FE5	:	.71705201262076255105948613570426
00000000	00000000	00007200	06C34000	0180	152	.OCTA	^X00000000000000000000720096C34000	:	.79445987939834594726562500000000
				0190	153	: Entry 5			
00000000	00000000	0000A400	33174001	0190	154	.OCTA	^X00000000000000000000A40033174001	:	.11995794773101806640625000000000
4BF2704B	F5DA0C33	2C1797A2	35F73FE7	01A0	155	.OCTA	^X4BF2704BF5DA0C332C1797A235F73FE7	:	.18042463197685219415338439146405
0000D2A0	6E1BE979	6AD43BC9	74AD3FFE	01B0	156	.OCTA	^X0000D2A06E1BE9796AD43BC974AD3FFE	:	.18197104175974705953372566179626
67754F6A	EF080C33	2C1797A2	35F73FE7	01C0	157	.OCTA	^X67754F6AEF080C332C1797A235F73FE7	:	.18042463197685219415316916451793
00000000	00000000	0000F600	AAD04000	01D0	158	.OCTA	^X00000000000000000000F600AAD04000	:	.83362549543380737304687500000000
				01E0	159	: Entry 6			
00000000	00000000	00004C00	27FF4001	01E0	160	.OCTA	^X000000000000000000004C0027FF4001	:	.11562392711639404296875000000000
E508D66E	AD727307	A85B7F52	A54F3FE7	01F0	161	.OCTA	^XE508D66EAD727307A85B7F52A54F3FE7	:	.24523500850365411600934046288264
00003DE0	E3DA3E19	1D014ECE	29503FFE	0200	162	.OCTA	^X00003DE0E3DA3E191D014ECE29503FFE	:	.14517270628459810425716671490628
7A566B68	A1627307	A85B7F52	A54F3FE7	0210	163	.OCTA	^X7A566B68A1627307A85B7F52A54F3FE7	:	.24523500850365411600895978452072
00000000	00000000	0000A000	BAD04000	0220	164	.OCTA	^X00000000000000000000A000BAD04000	:	.86487293243408203125000000000000
				0230	165	: Entry 7			
00000000	00000000	00001800	1F834001	0230	166	.OCTA	^X0000000000000000000018001F834001	:	.11230940818786621093750000000000
AC20ED9E	23171083	3B0FDFD8	1D633FE6	0240	167	.OCTA	^XAC20ED9E231710833B0FDFD81D633FE6	:	.83059460840978111968082197259804
00007FF0	9A9A38F9	EE168137	DB7E3FFD	0250	168	.OCTA	^X00007FF09A9A38F9EE168137DB7E3FFD	:	.11608744121520469473521987338837
FAAB19A9	1E4D1083	3B0FDFD8	1D633FE6	0260	169	.OCTA	^XFAAB19A91E4D10833B0FDFD81D633FE6	:	.83059460840978111968006588149715
00000000	00000000	00002A00	C7E24000	0270	170	.OCTA	^X000000000000000000002A00C7E24000	:	.89039736986160278320312500000000
				0280	171	: Entry 8			
00000000	00000000	00007200	18B64001	0280	172	.OCTA	^X00000000000000000000720018B64001	:	.10965338945388793945312500000000
6C77FE6B	932537A6	18911C7B	6309BFE7	0290	173	.OCTA	^X6C77FE6B932537A618911C7B6309BFE7	:	.20665791283430593743887663719157
00005350	408AF71C	1ED7B43D	79763FFD	02A0	174	.OCTA	^X00005350408AF71C1ED7B43D79763FFD	:	.92154220636009746064863489815427
A32DE46E	96AD37A6	18911C7B	6309BFE7	02B0	175	.OCTA	^XA32DE46E96AD37A618911C7B6309BFE7	:	.20665791283430593743898805128193
00000000	00000000	00000200	D2ED4000	02C0	176	.OCTA	^X000000000000000000000200D2ED4000	:	.91196447610855102539062500000000
				02D0	177	: Entry 9			
00000000	00000000	00006400	13654001	02D0	178	.OCTA	^X00000000000000000000640013654001	:	.10757658481597900390625000000000
E5C89455	C3E694FE	3DD96B68	16BF3FE8	02E0	179	.OCTA	^XE5C89455C3E694FE3DD96B6816BF3FE8	:	.32450507005410420602536543356999
000085F0	1FC3A187	79B76EEC	2B243FFD	02F0	180	.OCTA	^X000085F01FC3A18779B76EEC2B243FFD	:	.73032792373494277653040949442097
DD1F1F9A	C0D194FE	3DD96B68	16BF3FE8	0300	181	.OCTA	^XDD1F1F9AC0D194FE3DD96B6816BF3FE8	:	.32450507005410420602517081764557
00000000	00000000	0000A600	DBF04000	0310	182	.OCTA	^X00000000000000000000A600DBF04000	:	.92957037687301635742187500000000
				0320	183	: Entry 10			
00000000	00000000	0000BE00	0F694001	0320	184	.OCTA	^X00000000000000000000BE000F694001	:	.10602072477340698242187500000000
FFFA4F92	524356DF	50072B79	73603FE7	0330	185	.OCTA	^XFFFA4F92524356DF50072B7973603FE7	:	.21616908684298677364717595414676
00002540	34AD9102	B83FB339	DEF03FFC	0340	186	.OCTA	^X0000254034AD9102B83FB339DEF03FFC	:	.58464384126416698177476064883045
3632E125	4F5F56DF	50072B79	73603FE7	0350	187	.OCTA	^X3632E1254F5F56DF50072B7973603FE7	:	.21616908684298677364708481219455
00000000	00000000	0000AA00	E2EC4000	0360	188	.OCTA	^X00000000000000000000AA00E2EC4000	:	.94321185350418090820312500000000
				0370	189	: Entry 11			
00000000	00000000	00004800	0CA84001	0370	190	.OCTA	^X0000000000000000000048000CA84001	:	.10494427680969238281250000000000
F590F833	C5BE6CF4	202D00A8	2565BFE8	0380	191	.OCTA	^XF590F833C5BE6CF4202D00A82565BFE8	:	.34155619944337780728715701986708
00005DF0	98C41045	49EE36D5	8B573FFC	0390	192	.OCTA	^X00005DF098C4104549EE36D58B573FFC	:	.48259360404981917320950354395334
82E966B3	C5EE6CF4	202D00A8	2565BFE8	03A0	193	.OCTA	^X82E966B3C5EE6CF4202D00A82565BFE8	:	.34155619944337780728716871266757


```
00000000 00000000 0000C000 E7E04000 03B0 194 .OCTA ^X00000000000000000000C000E7E04000 ; .95288658142089843750000000000000
03C0 195 ; Entry 12
00000000 00000000 00009200 0A704001 03C0 196 .OCTA ^X00000000000000000000092000A704001 ; .10407801866531372070312500000000
76DA2D8A 21E235C8 AE83AFC8 50963FE6 03D0 197 .OCTA ^X76DA2D8A21E235C8AE83AFC850963FE6 ; .97960181228597287394528374900967
00000130 6604EF53 D39A6D53 47703FFC 03E0 198 .OCTA ^X000001306604EF53D39A6D5347703FFC ; .39970601587458680066945470960825
38C557A8 1F4D35C8 AE83AFC8 50963FE6 03F0 199 .OCTA ^X38C557A81F4D35C8AE83AFC850963FE6 ; .97960181228597287394487647769899
00000000 00000000 00004C00 EBF04000 0400 200 .OCTA ^X0000000000000000000004C00EBF04000 ; .96081769466400146484375000000000
0410 201 ; Entry 13
00000000 00000000 0000D200 08DD4001 0410 202 .OCTA ^X00000000000000000000D20008DD4001 ; .10346347093582153320312500000000
7ADBFEA1 9608511B 0185CFB4 8A813FE6 0420 203 .OCTA ^X7ADBFEA19608511B0185CFB48A813FE6 ; .11481667040746137460866705622794
000068D0 D46C9834 83D9B3AD 16EC3FFC 0430 204 .OCTA ^X000068D0D46C983483D9B3AD16EC3FFC ; .34048415120305718879047780165238
25435652 8FE4511B 0185CFB4 8A813FE6 0440 205 .OCTA ^X254356528FE4511B0185CFB48A813FE6 ; .11481667040746137460857013372888
00000000 00000000 00005400 EEDC4000 0450 206 .OCTA ^X0000000000000000000005400EEDC4000 ; .96652472019195556640625000000000
0460 207
0460 208 ; Polynomial constants tables
0460 209
0460 210
0460 211
0460 212
0460 213 LOGTAB1: ; Constants for q(z). Generated using eq. 6.3.10 of Hart et.
0460 214 ; al. (sin(2a) = 1/32)
0460 215
5F95F1B2 A5BAC3D8 F5260A61 9B9BBFFC 0460 216 .OCTA ^X5F95F1B2A5BAC3D8F5260A619B9BBFFC ; C19 = -.50244827536208487853089580
92DBE7A6 76579059 6C52CC82 B1293FFC 0470 217 .OCTA ^X92DBE7A6765790596C52CC82B1293FFC ; C18 = -.52876376564549972132965432
7A54AC14 946576FF 14A540A3 C71BBFFC 0480 218 .OCTA ^X7A54AC14946576FF14A540A3C71BBFFC ; C17 = -.55554987186628930743974171
5E0B06A1 8F25A3D8 4658C0D9 E1E03FFC 0490 219 .OCTA ^X5E0B06A18F25A3D84658C0D9E1E03FFC ; C16 = -.58822991044688315301145533
AE7E786D A6F3EE3D 371F0033 0000BFFD 04A0 220 .OCTA ^XAE7E786DA6F3EE3D371F00330000BFFD ; C15 = -.62500000745281154707785818
4B4434C7 0327C803 9543113E 11113FFD 04B0 221 .OCTA ^X4B4434C70327C8039543113E11113FFD ; C14 = -.66666667329017444142627970
298C1180 F148E440 879C4924 2492BFFD 04C0 222 .OCTA ^X298C1180F148E440879C49242492BFFD ; C13 = -.71428571427964746924020644
E563A213 835C94C6 0AE7B13B 3B133FFD 04D0 223 .OCTA ^XE563A213835C94C60AE7B13B3B133FFD ; C12 = -.76923076922577400516904957
2CB2BF3F 9152C30A 55565555 5555BFFD 04E0 224 .OCTA ^X2CB2BF3F9152C30A555655555555BFFD ; C11 = -.833333333333333650534413512
0A06F262 E8796F50 D1751745 745D3FFD 04F0 225 .OCTA ^X0A06F262E8796F50D1751745745D3FFD ; C10 = -.9090909090909091146943301574
62840C66 2AF3997A 99999999 9999BFFD 0500 226 .OCTA ^X62840C662AF3997A999999999999BFFD ; C9 = -.9999999999999999999999893503466
FE968C1F 7E77C707 1C7171C7 C71C3FFD 0510 227 .OCTA ^XFE968C1F7E77C7071C7171C7C71C3FFD ; C8 = -.111111111111111111111104012828
ED6B8E90 00D70000 00000000 0000BFFE 0520 228 .OCTA ^XED6B8E9000D70000000000000000BFFE ; C7 = -.125000000000000000000000002228
47B3B871 499F2492 92494924 24923FFE 0530 229 .OCTA ^X47B3B871499F24929249492424923FFE ; C6 = -.14285714285714285714286987
BE5E4E98 55555555 55555555 5555BFFE 0540 230 .OCTA ^XBE5E4E98555555555555555555555BFFE ; C5 = -.166666666666666666666666666666
CEED967D 99999999 99999999 99993FFE 0550 231 .OCTA ^XCEED967D99999999999999999999993FFE ; C4 = -.199999999999999999999999999999
0DD20000 00000000 00000000 0000BFFF 0560 232 .OCTA ^X0DD2000000000000000000000000BFFF ; C3 = -.250000000000000000000000000000
59F05555 55555555 55555555 55553FFF 0570 233 .OCTA ^X59F055555555555555555555553FFF ; C2 = -.333333333333333333333333333333
00000000 00000000 00000000 0000C000 0580 234 .OCTA ^X00000000000000000000000000C000 ; C1 = -.500000000000000000000000000000
00000000 00000000 00000000 00000000 0590 235 .OCTA ^X000000000000000000000000000000 ; C0 = .000000000000000000000000000000
05A0 236
00000014 05A0 237 LOGLEN1 = -.LOGTAB1/16 ; no. of floating point entries
05A0 238
05A0 239
05A0 240 LOGTAB2: ; Constants for p(z*z). Generated using eq. 6.3.11 of Hart et.
05A0 241 ; al. (sin(2a) = (b - 1)/(b + 1) where b = 2**((1/7)))
05A0 242
8441440A 9DA42272 7A67F044 8B243FFD 05A0 243 .OCTA ^X8441440A9DA422727A67F0448B243FFD ; C10 = .9647077421655028896227926448
0019B5C5 3BD4BEC7 61F97E57 AF203FFD 05B0 244 .OCTA ^X0019B5C53BD4BEC761F97E57AF203FFD ; C9 = .1052555976112884972789729915
85B2D526 87082827 EEF2E8F7 E1E13FFD 05C0 245 .OCTA ^X85B2D52687082827EEF2E8F7E1E13FFD ; C8 = .1176470852214462141323328874
C286BAD2 232FAD44 1440110F 11113FFE 05D0 246 .OCTA ^XC286BAD2232FAD441440110F11113FFE ; C7 = .1333333332754878760888250485
90B321D3 AF744625 146BB13B 3B133FFE 05E0 247 .OCTA ^X90B321D3AF744625146BB13B3B133FFE ; C6 = .1538461538462364659507622054
F8411CEE 61EB3082 D1741745 745D3FFE 05F0 248 .OCTA ^XF8411CEE61EB3082D1741745745D3FFE ; C5 = .1818181818181817408450657725
73AA312A 4DE1C723 1C7171C7 C71C3FFE 0600 249 .OCTA ^X73AA312A4DE1C7231C7171C7C71C3FFE ; C4 = .22222222222222222222687058519
5FBA09F6 48D22492 92494924 24923FFF 0610 250 .OCTA ^X5FBA09F648D224929249492424923FFF ; C3 = .2857142857142857142856972183
```

[illegible]


```
0690 271      .SBTTL MTH$HLOG - Standard H-Floating LOG
0690 272
0690 273
0690 274 : FUNCTIONAL DESCRIPTION:
0690 275 :
0690 276 : HLOG - H floating point LOG function
0690 277 :
0690 278 : HLOG(X) is computed using the following approximation technique:
0690 279 :
0690 280 :   If X <= 0, error. Otherwise
0690 281 :
0690 282 :   Let X = f * (2**n), where 1/2 <= f < 1
0690 283 :
0690 284 :   If n is greater than or equal to 1 then
0690 285 :     set N = n - 1 and F = 2*f.
0690 286 :   Else
0690 287 :     set N = n and F = f.
0690 288 :
0690 289 :   Then ln(x) = N*ln2 + ln(F)
0690 290 :
0690 291 :   If |F - 1| < 2**-5 then
0690 292 :     ln(F) = W + W*P(W), where W = F - 1 and P
0690 293 :     is a polynomial of degree 18.
0690 294 :   Else
0690 295 :     ln(F) = ln(FHI) + Z*Q(Z*Z), where FHI is ob-
0690 296 :     tained by table look-up, Q is a polynomial of
0690 297 :     degree 10 and Z = (F - FHI)/(F + FHI)
0690 298 :
0690 299 :   NOTE: The quantities ln(FHI) and ln2 are used in the above
0690 300 :   equations in two parts - a high part (containing the
0690 301 :   high order bits) and a low part (containing the low
0690 302 :   order bits. In the code the high and low parts of the
0690 303 :   constants are indicated by a HI and LO suffix respec-
0690 304 :   tively. The values were chosen such that N*LN_2_HI +
0690 305 :   LN_FHI_HI is exactly representable.
0690 306 :
0690 307 : ++
0690 308 :
0690 309 : CALLING SEQUENCE:
0690 310 :
0690 311 :   hlog.wh.v = MTH$HLOG(x.rh.r)
0690 312 :   -or-
0690 313 :   CALL MTH$HLOG(hlog.wh.r, x.rh.r)
0690 314 :
0690 315 : INPUT PARAMETERS:
0690 316 :
0690 317 :   LONG = 4 ; Define longword multiplier
0690 318 :   x = 2 * LONG ; Contents of x is the argument
0690 319 :
0690 320 : IMPLICIT INPUTS: none
0690 321 :
0690 322 : OUTPUT PARAMETERS:
0690 323 :
0690 324 :   hlog = 1 * LONG ; Contents of hlog is the result
0690 325 :
0690 326 : VALUE: H floating logarithm of the argument
0690 327 :
```

00000004
00000008

00000004

```
0690 328 : IMPLICIT OUTPUTS: none
0690 329 :
0690 330 : COMPLETION CODES: none
0690 331 :
0690 332 : SIDE EFFECTS:
0690 333 :
0690 334 : Signals: MTH$_LOGZERNEG if !X! =< 0.0 with reserved operand in R0/R3 (copied
0690 335 : to the signal mechanism vector CHF$_MCH_R0/R1 by LIB$SIGNAL). Associated
0690 336 : message is: "LOGARITHM OF ZERO OR NEGATIVE VALUE". Result is reserved operand
0690 337 : -0.0 unless a user supplied (or any) error handler changes CHF$_MCH_R0/R1.
0690 338 :
0690 339 : NOTE: This procedure disables floating point underflow, enables integer over-
0690 340 : flow, causes no floating overflow or other arithmetic traps, and preserves
0690 341 : enables across the call.
0690 342 :
0690 343 : ---
0690 344 :
0690 345 :
41FC 0690 346 : .ENTRY MTH$HLOG, ACMASK ; Standard call-by-reference entry
0692 347 : ; Disable DV (and FU), enable IV
0692 348 : MTH$FLAG_JACKET ; Flag that this is a jacket procedure
0692 349 :
6D 00000000'GF 9E 0692 350 : MOVAB G^MTH$$JACKET_HND, (FP) ; set handler address to jacket
0699 351 : ; handler
0699 352 :
0699 353 : ; in case of an error in special JSB
0699 354 : ; routine
50 08 BC 70FD 0699 355 : MOVH @x(AP), R0 ; R0/R3 = arg
069E 356 : BSBB MTH$HLOG_R8 ; Call special HLOG routine
04 BC 50 7DFD 06A0 357 : MOVO R0, @hlog(AP) ; Store result in first argument
06A5 358 : RET ; Return to caller
06A6 359 :
```



```
06A6 357      .SBTTL MTH$HLOG10 - Standard H Floating Common logarithm
06A6 358
06A6 359      ;++
06A6 360      ; FUNCTIONAL DESCRIPTION:
06A6 361      ; HLOG10 - H floating point LOG10 function
06A6 362      ; HLOG10(X) is computed as HLOG10(E) * HLOG(X).
06A6 363      ;
06A6 364      ; See description of MTH$HLOG
06A6 365
06A6 366      ; CALLING SEQUENCE:
06A6 367      ;
06A6 368      ;     hlog_base_10.wh.v = MTH$HLOG10(x.rh.r)
06A6 369      ;
06A6 370      ;     -or-
06A6 371      ;     CALL MTH$HLOG10(hlog_base_10.wh.r, x.rh.r)
06A6 372
06A6 373      ; INPUT PARAMETERS:
06A6 374      ;
06A6 375      ;     LONG = 4                      ; Define longword multiplier
00000004 06A6 376      ; x = 2 * LONG                ; Contents of x is the argument
00000008 06A6 377
06A6 378      ;
06A6 379      ; OUTPUT PARAMETERS:
06A6 380      ;
06A6 381      ;     hlog_base_10 = 1 * LONG
00000004 06A6 382
06A6 383      ; SIDE EFFECTS: See description of MTH$HLOG
06A6 384      ;
06A6 385      ; --
06A6 386
06A6 387
06A6 388
06A6 389      .ENTRY MTH$HLOG10, ACMASK      ; Standard call-by-reference entry
06A8 390      MTH$FLAG_JACKET              ; Disable DV (and FU), enable IV
06A8 391      MOVAB G^MTH$$JACKET_HND, (FP) ; Flag that this is a jacket procedure
06A8
06AF      ; set handler address to jacket
06AF      ; handler
06AF
06AF      ; in case of an error in special JSB
06AF 392      ; routine
06AF 393      ; R0/R3 = arg
06AF 394      MOVH @x(AP), R0                ; Call special HLOG10 routine
06B4 395      BSBB MTH$HLOG10_R8          ; Store result in first argument
06B6 396      MOVO R0, @hlog_base_10(AP) ; Return to caller
06BB 397
06BC 398
```

6D 00000000'GF 9E

50 08 BC 70FD 1E 10

04 BC 50 7DFD 04

```
06BC 400      .SBTTL MTH$HLOG2 - Standard H Floating Common logarithm
06BC 401
06BC 402      ;++
06BC 403      ; FUNCTIONAL DESCRIPTION:
06BC 404
06BC 405      ; HLOG2 - H floating point LOG2 function
06BC 406
06BC 407      ; HLOG2(X) is computed as HLOG2(E) * HLOG(X).
06BC 408
06BC 409      ; See description of MTH$HLOG
06BC 410
06BC 411      ; CALLING SEQUENCE:
06BC 412
06BC 413      ;     hlog_base_2.wh.v = MTH$HLOG2(x.rh.r)
06BC 414      ;     -or-
06BC 415      ;     CALL MTH$HLOG2(hlog_base_2.wh.r, x.rh.r)
06BC 416
06BC 417      ; INPUT PARAMETERS:
06BC 418
00000004 06BC 419      ;     LONG = 4                ; Define longword multiplier
00000008 06BC 420      ;     x = 2 * LONG            ; Contents of x is the argument
06BC 421
06BC 422
06BC 423      ; OUTPUT PARAMETERS:
06BC 424
00000004 06BC 425      ;     hlog_base_2 = 1 * LONG
06BC 426
06BC 427      ; SIDE EFFECTS: See description of MTH$HLOG
06BC 428
06BC 429      ;--
06BC 430
06BC 431
41FC 06BC 432      .ENTRY MTH$HLOG2, ACMASK      ; Standard call-by-reference entry
06BE 433      ; Disable DV (and FU), enable IV
06BE 434      ; MTH$FLAG_JACKET                ; Flag that this is a jacket procedure
06BE
6D 00000000'GF 9E 06BE      MOVAB G^MTH$$JACKET_HND, (FP)
06C5      ; set handler address to jacket
06C5      ; handler
06C5
06C5 435      ; in case of an error in special JSB
06C5 436      ; routine
06C5 437      MOVH @x(AP), R0                ; R0/R3 = arg
06CA 438      BSBB MTH$HLOG_R8            ; calculate natural log
04 BC 50 B0 AF 65FD 06CC 439      MULH3 H_INV_LN2_CONS, R0, @hlog_base_2(AP)
06D3 440      ; convert and store result in first
06D3 441      ; argument
04 06D3 442      RET                    ; Return to caller
```



```
06D4 444 .SBTTL MTH$HLOGHLOG10_R8 - Special HLOG/HLOG10 routines
06D4 445
06D4 446 ; Special HLOG/HLOG10 - used by the standard routine, and directly.
06D4 447
06D4 448 ; CALLING SEQUENCE:
06D4 449 ; save anything needed in R0:R8
06D4 450 ; MOVH R0 ; Input in R0/R3
06D4 451 ; JSB MTH$HLOG10_R8 /MTH$HLOG_R8
06D4 452 ; return with result in R0/R3
06D4 453 ; Note: This routine is written to avoid causing any integer overflows,
06D4 454 ; floating overflows, or floating underflows or divide by 0 conditions,
06D4 455 ; whether enabled or not.
06D4 456
06D4 457 ; REGISTERS USED:
06D4 458 ; R0/R3 - H floating argument then result
06D4 459 ; R4/R7 - Intermediate results
06D4 460 ; R0:R5 - POLYH
06D4 461 ; R8 - Pointer into H_FHI table
06D4 462
06D4 463
06D4 464
06D4 465 MTH$HLOG10_R8::
57 50 B0 06D4 466 MOVW R0, R7 ; Special HLOG10 routine
08 15 06D7 467 BLEQ ERR ; R7 = biased exponent
06D9 468 ; Error if <= 0
06D9 469 ; User PC on top of stack
06D9 470 ; Note: ERROR routine depends on user
06D9 471 ; PC being on top of stack, so
06D9 472 ; subroutine call to MTH$HLOG_R8 is not
06D9 473 ; used.
50 0E 10 06D9 474 BSBB HLOG_COMMON_R8 ; Call common HLOG/HLOG10 routine
91 AF 64FD 06DB 474 MULH2 H_LOG10_E, R0 ; R0/R3 = LOG10(e) * LOG(X)
05 06E0 475 RSB ; Return
06E1 476
06E1 477 ERR: BRW ERROR
06E4 478
06E4 479 MTH$HLOG_R8::
57 50 B0 06E4 480 MOVW R0, R7 ; special LOG routine
F8 15 06E7 481 BLEQ ERR ; R7 = Biased exponent
06E9 482 HLOG_COMMON_R8: ; HLOG(X) is not defined for X=<0
57 4000 8F A2 06E9 483 SUBW #^X4000, R7 ; R7 = Unbiased exponent
6A 15 06EE 484 BLEQ NEG_EXP ; Branch to processing for n=<0
06F0 485
06F0 486 ;
06F0 487 ; Exponent is positive. N = n - 1 and F = 2f
06F0 488 ;
06F0 489
06F0 490 DECW R7 ; R7 = N = n - 1
58 50 57 B7 06F0 491 SUBW R7, R0 ; R0/R3 = F = 2f
58 50 07 9C 06F2 491 ROTL #7, R0, R8 ; R8 = index into MTH$$AB ALOG table
58 FFFFFFF0 8F CA 06F9 493 BICL #-256, R8 ; = lo exp bit and 1st 7 fract bits
56 00000000 GF DE 0700 494 MOVAL G^MTH$$AB ALOG_V, R6 ; R6 = Address of RTL vector entry
56 56 66 C0 0707 495 ADDL (R6), R6 ; R6 = Address of MTH$$AB ALOG table
58 6648 90 070A 496 MOVW (R6)[R8], R8 ; R8 = offset into H_FHI tables
48 19 070E 497 BLSS LN_1_PLUS ; Branch to special processing
0710 498 ; for F close to 1
0710 499
0710 500 ;
```

```
0710 501 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
0710 502 ;
0710 503 ;
58 7E 57 6DFD 0710 504 CVTWH R7, -(SP) ; Push N onto the stack
54 F8E6 CF48 7EFD 0710 505 MOVAB MTH$AB_H_FHI[R8], R8 ; R8 = Address of FHI
50 50 68 63FD 0714 506 SUBH3 (R8), R0, R4 ; R4/R7 = F - FHI
50 50 88 60FD 071B 507 ADDH2 (R8), R0 ; R0/R3 = F + FHI
7E 54 50 67FD 0720 508 DIVH3 R0, R4, -(SP) ; (SP) = Z = (F - FHI)/(F + FHI)
50 6E 6E 65FD 0724 509 MULH3 (SP), (SP), R0 ; R0/R3 = Z**2
FE6B CF 0A 50 75FD 0729 510 POLYH R0, #LOGLEN2-1, LOGTAB2 ; R0/R3 = P(Z**2)
50 8E 64FD 072E 511 MULH2 (SP)+, R0 ; R0/R3 = Z*P(Z**2)
0735 512 ;
0739 513 ;
0739 514 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z**2)
0739 515 ;
54 FF21 CF 6E 65FD 0739 516 MULH3 (SP), H_LN_2_LO, R4 ; R4/R7 = N*LN_2_LO
54 54 88 60FD 0740 517 ADDH2 (R8)+, R4 ; R4/R7 = N*LN_2_LO + LN_FHI_LO
50 54 60FD 0744 518 ADDH2 R4, R0 ; R0/R3 = B
0748 519 ;
0748 520 ;
0748 521 ; Compute A = N*LN_2_HI + LN_FHI_HI and HLOG(X)
0748 522 ;
54 FF02 CF 8E 65FD 0748 523 MULH3 (SP)+, H_LN_2_HI, R4 ; R4/R7 = N*LN_2_HI
54 54 68 60FD 074F 524 ADDH2 (R8), R4 ; R4/R7 = A = N*LN_2_HI + LN_FHI_HI
50 54 60FD 0753 525 ADDH2 R4, R0 ; R0/R3 = A + B = HLOG(X)
0757 526 RSB
0758 527 ;
0758 528 LN_1_PLUS:
66 11 0758 529 BRB LN_1_PLUS_W
075A 530 ;
075A 531 ;
075A 532 ;
075A 533 ; Exponent is negative. N = n and F = f
075A 534 ;
075A 535 ;
50 57 A2 075A 536 NEG_EXP: SUBW R7, R0 ; R0/R3 = F = f
58 50 07 9C 075D 537 ROTL #7, R0, R8 ; R8 = index into MTH$AB ALOG table
58 FFFFFFF0 8F CA 0761 538 BICL #-256, R8 ; = lo exp bit and 1st 7 fract bits
56 00000000 GF DE 0768 539 MOVAB MTH$AB ALOG_V, R6 ; R6 = Address of RTL vector entry
56 56 66 C0 076F 540 ADDL (R6), R6 ; R6 = Address of MTH$AB ALOG table
58 66 48 90 0772 541 MOVAB (R6)[R8], R8 ; R8 = offset into H_FHI tables
48 19 0776 542 BLSS LN_1_PLUS_W ; Branch to special processing
0778 543 ; for F close to 1
0778 544 ;
0778 545 ;
0778 546 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
0778 547 ;
0778 548 ;
58 7E 57 6DFD 0778 549 CVTWH R7, -(SP) ; Push N onto the stack
54 F87E CF48 7EFD 077C 550 MOVAB MTH$AB_H_FHI[R8], R8 ; R8 = Address of FHI
50 50 68 63FD 0783 551 SUBH3 (R8), R0, R4 ; R4/R7 = F - FHI
50 50 88 60FD 0788 552 ADDH2 (R8), R0 ; R0/R3 = F + FHI
7E 54 50 67FD 078C 553 DIVH3 R0, R4, -(SP) ; (SP) = Z = (F - FHI)/(F + FHI)
50 6E 6E 65FD 0791 554 MULH3 (SP), (SP), R0 ; R0/R3 = Z**2
FE03 CF 0A 50 75FD 0796 555 POLYH R0, #LOGLEN2-1, LOGTAB2 ; R0/R3 = P(Z**2)
50 8E 64FD 079D 556 MULH2 (SP)+, R0 ; R0/R3 = Z*P(Z**2)
07A1 557 ;
```



```
07A1 558 :  
07A1 559 : Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)  
07A1 560 :  
54 FEB9 CF 6E 65FD 07A1 561 MULH3 (SP), H LN_2_LO, R4 ; R4/R7 = N*LN_2_LO  
54 54 78 60FD 07A8 562 ADDH2 -(R8), R4 ; R4/R7 = N*LN_2_LO + LN_FHI_LO  
50 54 60FD 07AC 563 ADDH2 R4, R0 ; R0/R3 = B  
07B0 564 :  
07B0 565 :  
07B0 566 : Compute A = N*LN_2_HI + LN_FHI_HI and HLOG(X)  
07B0 567 :  
54 FE9A CF 8E 65FD 07B0 568 MULH3 (SP)+, H LN_2_HI, R4 ; R4/R7 = N*LN_2_HI  
54 54 78 62FD 07B7 569 SUBH2 -(R8), R4 ; R4/R7 = A = N*LN_2_HI + LN_FHI_HI  
50 54 60FD 07BB 570 ADDH2 R4, R0 ; R0/R3 = A + B = HLOG(X)  
05 07BF 571 RSB  
07C0 572 :  
07C0 573 :  
07C0 574 : Special logic for F close to 1  
07C0 575 :  
07C0 576 :  
07C0 577 LN_1_PLUS W:  
FC94 7E 50 08 63FD 07C0 578 SUBH3 #1, R0, -(SP) ; (SP) = W = F - 1  
CF 13 6E 75FD 07C5 579 POLYH (SP), #LOGLEN1-1, LOGTAB1 ; R0/R3 = Q(W)  
50 6E 64FD 07CC 580 MULH2 (SP), R0 ; R0/R3 = W*Q(W)  
54 57 60FD 07D0 581 CVTWH R7, R4 ; R4/R5 = N  
7E FE86 CF 54 65FD 07D4 582 MULH3 R4, H LN_2_LO, -(SP) ; (SP) = N*LN_2_LO  
50 8E 60FD 07DB 583 ADDH2 (SP)+, R0 ; R0/R3 = N*LN_2_LO + W*Q(W)  
50 8E 60FD 07DF 584 ADDH2 (SP)+, R0 ; R0/R3 = N*LN_2_LO + LN(F)  
54 FE68 CF 64FD 07E3 585 MULH2 H LN_2_HI, R4 ; R4/R5 = N*LN_2_HI  
50 54 60FD 07E9 586 ADDH2 R4, R0 ; R0/R3 = HLOG(X)  
05 07ED 587 RSB  
07EE 588 :  
07EE 589 : x =< 0.0, signal error  
07EE 590 :  
07EE 591 ERROR: PUSHL (SP) ; Return PC from JSB routine  
50 7E 00 8F 9A 07F0 592 MOVZBL #MTH$K_LOGZERNEG, -(SP) ; Condition value  
50 01 0F 79 07F4 593 ASHQ #15, #T, R0 ; R0 = result = reserved operand -0.0  
07F8 594 : Goes to signal mechanism vector  
07F8 595 : (CHF$MCH_R0/R3) so error handler  
07F8 596 : Can modify the result.  
00000000'GF 52 7C 07F8 597 CLRQ R2 ;  
02 FB 07FA 598 CALLS #2, G^MTH$$SIGNAL ; Signal error and use real user's PC  
05 0801 599 : Independent of CALL vs JSB  
0801 600 RSB ; Return - R0 restored from CHF$MCH_R0/R3  
0802 601 :  
0802 602 :  
0802 603 :  
0802 604 .END
```

MTH\$HLOG
Symbol table

1 3
; Floating Point Natural and Common

16-SEP-1984 01:36:48 VAX/VMS Macro V04-00
6-SEP-1984 11:25:02 [MTHRTL.SRC]MTHHLOG.MAR;1

Page 15
(7)

ACMASK = 000041FC
ERR 000006E1 R 01
ERROR 000007EE R 01
HLOG = 00000004
HLOG_BASE_10 = 00000004
HLOG_BASE_2 = 00000004
HLOG_COMMON_R8 000006E9 R 01
H_INV_LN2_CONS 00000680 R 01
H_LN_2_HI 00000650 R 01
H_LN_2_LO 00000660 R 01
H_LOG10_E 00000670 R 01
LN_1_PLUS 00000758 R 01
LN_1_PLUS_W 000007C0 R 01
LOGLEN1 = 00000014
LOGLEN2 = 00000008
LOGTAB1 00000460 R 01
LOGTAB2 000005A0 R 01
LONG = 00000004
MTH\$\$AB ALOG_V ***** X 00
MTH\$\$AB_H_FHT 00000000 RG 01
MTH\$\$JACKET_HND ***** X 01
MTH\$\$SIGNAL ***** X 00
MTH\$HLOG 00000690 RG 01
MTH\$HLOG10 000006A6 RG 01
MTH\$HLOG10_R8 000006D4 RG 01
MTH\$HLOG2 000006BC RG 01
MTH\$HLOG_R8 000006E4 RG 01
MTH\$K LOGZERNEG ***** X 00
NEG_EXP 0000075A R 01
X = 00000008

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes													
ABS	00000000 (0.)	00 (0.)	NOPI	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE			
MTH\$CODE	00000802 (2050.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG			

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.63
Command processing	110	00:00:00.58	00:00:03.64
Pass 1	106	00:00:01.83	00:00:05.48
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	126	00:00:01.39	00:00:06.46
Symbol table output	4	00:00:00.04	00:00:00.04
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	380	00:00:03.95	00:00:16.28

The working set limit was 1050 pages.

11062 bytes (22 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 30 non-local and 0 local symbols.
664 source lines were read in Pass 1, producing 20 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

+-----+
! Macro library statistics !
+-----+

Macro library name

Macros defined

_S255\$DUA28:[SYSLIB]STARLET.MLB;2

0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:MTHHLOG/OBJ=OBJ\$:MTHHLOG MSRC\$:MTHJACKET/UPDATE=(ENH\$:MTHJACKET)+MSRC\$:

0262 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY